Python: module browser.gui_alter_plot

browser.gui_alter_plot

index

```
# The PCMDI Data Browser Alter Notebook Popup -  gui_alter_plot (PDB) module
#
###############################################################################
#                                                                             #
# Module:       gui_alter_plot notebook module                                #
#                                                                             #
# Copyright:    "See file Legal.htm for copyright information."               #
#                                                                             #
# Authors:      PCMDI Software Team                                           #
#               Lawrence Livermore NationalLaboratory:                        #
#               support@pcmdi.llnl.gov                                        #
#                                                                             #
# Description:  GUI popup dialog to alter the appearance of the VCS plot      #
#                                                                             #
# Version:      4.0                                                           #
#                                                                             #
###############################################################################
#
#---------------------------------------------------------------------
# NOTE: need to use version of Python that imports Tkinter and Pmw
#---------------------------------------------------------------------
```

Modules

Numeric	browser.gui_control	sys
Tkinter	os	types
gui_support.gui_color	string	browser.vcs_function

Classes

note_book

class ***note_book***

```
#---------------------------------------------------------------------
# Popup to alter the visual appearance of the VCS plot.
#---------------------------------------------------------------------
```

Methods defined here:

__init__(self, parent)

apply(self, parent)

cbn10_cb(self)

cbn11_cb(self)

cbn12_cb(self)

cbn13_cb(self)

cbn14_cb(self)

cbn15_cb(self)

cbn16_cb(self)

cbn17_cb(self)

cbn18_cb(self)

cbn19_cb(self)

cbn1_cb(self)

cbn20_cb(self)

cbn21_cb(self)

cbn22_cb(self)

cbn2_cb(self)

cbn3_cb(self)

cbn4_cb(self)

cbn5_cb(self)

cbn6_cb(self)

cbn7_cb(self)

cbn8_cb(self)

```
major_log_axis, minor_log_axis = generate_log_values()
try:
   self.eny1.setentry( major_log_axis )
   self.eny2.setentry( minor_log_axis )
except:
   pass
self.parent.graphics_method.xticlabels1 = major_log_axis
```

```
        self.parent.graphics_method.xticlabels2 = major_log_axis
        self.parent.graphics_method.xmtics1 = minor_log_axis
        self.parent.graphics_method.xmtics2 = minor_log_axis
```

cbn9_cb(self)

```
        major_sine_axis, minor_sine_axis = generate_sine_values()
        try:
           self.eny1.setentry( major_sine_axis )
           self.eny2.setentry( minor_sine_axis )
        except:
           pass
        self.parent.graphics_method.xticlabels1 = major_sine_axis
        self.parent.graphics_method.xticlabels2 = major_sine_axis
        self.parent.graphics_method.xmtics1 = minor_sine_axis
        self.parent.graphics_method.xmtics2 = minor_sine_axis
```

cbnAwt1_cb(self)

cbnAwt2_cb(self)

cbnLn1_cb(self)

cbnLn2_cb(self)

cbnstat_off_cb(self)

cbnstat_on_cb(self)

execute(self, parent, result)

get_legend_settings(self)

get_plot_settings(self)

get_template_settings(self)

get_x_settings(self)

get_x_y_axis_entries(self)

get_y_settings(self)

hold_alter_original_attr_settings(self, parent)

reset_alter_to_original_settings(self, parent)

set_default_xlabels(self)

set_default_ylabels(self)

Functions

generate_log_values()
```
# Generate log10 dictionary values
```

generate_sine_values()
```
# Generate sine dictionary values
```

generate_x_axis(parent, graphics_method, replot_flg)
```
#--------------------------------------------------------------------------
# Set the graphics methods x-axis according to the "alter plot" po
#--------------------------------------------------------------------------
```

generate_y_axis(parent, graphics_method, replot_flg)
```
#--------------------------------------------------------------------------
# Set the graphics methods y-axis according to the "alter plot" po
#--------------------------------------------------------------------------
```

initialize(parent)
```
#--------------------------------------------------------------------------
# Initialize the alter plot's default settings
#--------------------------------------------------------------------------
```

settings(parent, replot_flg, g_name, template_name)
```
#   if graphics_method.yaxisconvert == 'log10':
#      major_log_axis={}
#      minor_log_axis={}
#      a=10000.0
#      b=a/10.0
#      for j in range(8):
#       for i in range (a/b):
#        log_num = Numeric.log10(a - (i*b))
#        minor_log_axis[log_num] = str(a - (i*b))
#        if ( log_num - int(log_num) ) == 0:
#         major_log_axis[log_num] = str(a - (i*b))
#       a=b
#       b=a/10.0
#      #
#      graphics_method.yticlabels1 = major_log_axis
#      graphics_method.ymtics1 = minor_log_axis
#      graphics_method.yticlabels2=major_log_axis
#      graphics_method.ymtics2 = minor_log_axis
#   elif replot_flg != 3:
#      graphics_method.yticlabels1 = '*'
#      graphics_method.ymtics1 = '*'
#      graphics_method.yticlabels2 = '*'
#      graphics_method.ymtics2 = '*'
#   else:
#      major_str = parent.alter_notebook.eny3.get()
#      minor_str = parent.alter_notebook.eny4.get()
#      graphics_method.yticlabels1 = '*'
#       if (major_str != '*') and (major_str != ''):
#          s=eval(major_str)
#          graphics_method.yticlabels1 = s
#       graphics_method.ymtics1 = ''
#       if (minor_str != '*') and (minor_str != ''):
#          s=eval(minor_str)
#          graphics_method.ymtics1 = s
#       graphics_method.yticlabels2 = '*'
#       graphics_method.ymtics2 = ''
#
#
#-----------------------------------------------------------------
# Set the graphics methods according to the "alter plot" popup con
#-----------------------------------------------------------------
```

write_legend_page(self, page)

```
#-----------------------------------------------------------------
# Write the "Legend" page to the notebook
#-----------------------------------------------------------------
```

write_shape_page(self, page)

```
#-----------------------------------------------------------------
# Write the "Shape" page of the notebook
#-----------------------------------------------------------------
```

write_template_page(self, page)

```
#-----------------------------------------------------------------
# Write the "Template" page of the notebook.
#-----------------------------------------------------------------
```

write_x_axis_page(self, page)

```
#-----------------------------------------------------------------
# Write the "X-Axis" page of the notebook.
#-----------------------------------------------------------------
```

write_y_axis_page(self, page)

```
#-----------------------------------------------------------------
# Write the "Y-Axis" page to the notebook
#-----------------------------------------------------------------
```

Data

Pmw = <Pmw.Pmw_1_2.lib.PmwLoader.PmwLoader instance>
fn = '/pcmdi/halliday1/PCMDI_GRAPHICS'
global_xmtics = '*'
global_xticlabels = '*'
global_ymtics = '*'
global_yticlabels = '*'